Laporan Resmi Praktikum Pengolahan Citra

Dimas Rizky H.P. – 2110141011

3 D4 IT A

1. Thinning citra – Source Code mesthin.m

function Testthin

IterThinning = 100 ;

Image = imread('bw2.bmp') ;

if ndims(Image) == 0

Image = im2bw( Image ) ;

end

Raw = Image ;

for Iter = 1:IterThinning

OutBW1 = Condition1( Image, 0 ) ;

OutBW2 = Condition2( OutBW1, 0 ) ;

Image = OutBW2 ;

end

%% debug and compare the result in Matlab

I = bwmorph(Raw, 'thin',IterThinning );

close all ;

figure,imshow( Raw ) ;

figure,imshow( I ) ;

figure,imshow( OutBW2 ) ;

* Condition1.m

function OutBW = Condition1( BW , debug )

MaskBW = zeros(size(BW)+2) ;

MaskBW([2:end-1],[2:end-1]) = BW ;

BW = MaskBW ;

BW = double( BW ) ;

OutBW = BW ;

[row col] = find( BW == 1 ) ;

for i = 1 : length(row)

indx = row(i) ;

indy = col(i) ;

if debug == 1

if indx == 189 && indy == 28

close all ;

imshow(BW) ;

hold on ;

plot(indy,indx,'r\*') ;

end

end

Mask = [ BW(indx-1,indy-1) BW(indx-1,indy) BW(indx-1,indy+1) ;...

BW(indx,indy-1 ) BW(indx,indy) BW(indx,indy+1 ) ;...

BW(indx+1,indy-1) BW(indx+1,indy) BW(indx+1,indy+1) ];

%% Condition 1

if Mask(2,3) == 0 && ( Mask(1,3) == 1 || Mask(1,2) == 1)

b1 = 1;

else

b1 = 0;

end

if Mask(1,2) == 0 && ( Mask(1,1) == 1 || Mask(2,1) == 1)

b2 = 1;

else

b2 = 0;

end

if Mask(2,1) == 0 && ( Mask(3,1) == 1 || Mask(3,2) == 1)

b3 = 1;

else

b3 = 0;

end

if Mask(3,2) == 0 && ( Mask(3,3) == 1 || Mask(2,3) == 1)

b4 = 1;

else

b4 = 0;

end

b = b1 + b2 + b3 + b4 ;

%% Condition 2

Mask = logical(Mask) ;

N1 = double( Mask(2,3) | Mask(1,3) ) + double( Mask(1,2) | Mask(1,1) ) + double( Mask(2,1) | Mask(3,1) ) + double( Mask(3,2) | Mask(3,3) );

N2 = double( Mask(1,3) | Mask(1,2) ) + double( Mask(1,1) | Mask(2,1) ) + double( Mask(3,1) | Mask(3,2) ) + double( Mask(3,3) | Mask(2,3) ) ;

NCondition2 = min(N1,N2) ;

%% Condition 3

Mask = logical(Mask) ;

Mask(3,3) = ~Mask(3,3) ;

NCondition3 = ( Mask(1,3) | Mask(1,2) | Mask(3,3) ) & Mask(2,3) ;

%% Judgement

if NCondition3 == 0 && NCondition2>=2 && NCondition2<=3 && b == 1

OutBW(indx,indy) = 0 ;

end

end

OutBW = OutBW([2:end-1],[2:end-1]) ;

* + Condition2.m

function OutBW = Condition2( BW , debug )

MaskBW = zeros(size(BW)+2) ;

MaskBW([2:end-1],[2:end-1]) = BW ;

BW = MaskBW ;

BW = double( BW ) ;

OutBW = BW ;

[row col] = find( BW == 1 ) ;

for i = 1 : length(row)

indx = row(i) ;

indy = col(i) ;

if debug == 1

imshow(BW) ;

if indx == 27 && indy == 188

hold on ;

plot(indy,indx,'r\*') ;

end

end

Mask = [ BW(indx-1,indy-1) BW(indx-1,indy) BW(indx-1,indy+1) ;...

BW(indx,indy-1 ) BW(indx,indy) BW(indx,indy+1 ) ;...

BW(indx+1,indy-1) BW(indx+1,indy) BW(indx+1,indy+1) ];

%% Condition 1

if Mask(2,3) == 0 && ( Mask(1,3) == 1 || Mask(1,2) == 1)

b1 = 1;

else

b1 = 0;

end

if Mask(1,2) == 0 && ( Mask(1,1) == 1 || Mask(2,1) == 1)

b2 = 1;

else

b2 = 0;

end

if Mask(2,1) == 0 && ( Mask(3,1) == 1 || Mask(3,2) == 1)

b3 = 1;

else

b3 = 0;

end

if Mask(3,2) == 0 && ( Mask(3,3) == 1 || Mask(2,3) == 1)

b4 = 1;

else

b4 = 0;

end

b = b1 + b2 + b3 + b4 ;

%% Condition 2

Mask = logical(Mask) ;

N1 = double( Mask(2,3) | Mask(1,3) ) + double( Mask(1,2) | Mask(1,1) ) + double( Mask(2,1) | Mask(3,1) ) + double( Mask(3,2) | Mask(3,3) );

N2 = double( Mask(1,3) | Mask(1,2) ) + double( Mask(1,1) | Mask(2,1) ) + double( Mask(3,1) | Mask(3,2) ) + double( Mask(3,3) | Mask(2,3) ) ;

NCondition2 = min(N1,N2) ;

%% Condition New3

Mask = logical(Mask) ;

Mask(1,1) = ~Mask(1,1) ;

NCondition3 = ( Mask(3,1) | Mask(3,2) | Mask(1,1) ) & Mask(2,1);

%% Judgement

if NCondition3 == 0 && NCondition2>=2 && NCondition2<=3 && b == 1

OutBW(indx,indy) = 0 ;

end

end

OutBW = OutBW([2:end-1],[2:end-1]) ;

* Condition2.m

function OutBW = Condition2( BW , debug )

MaskBW = zeros(size(BW)+2) ;

MaskBW([2:end-1],[2:end-1]) = BW ;

BW = MaskBW ;

BW = double( BW ) ;

OutBW = BW ;

[row col] = find( BW == 1 ) ;

for i = 1 : length(row)

indx = row(i) ;

indy = col(i) ;

if debug == 1

imshow(BW) ;

if indx == 27 && indy == 188

hold on ;

plot(indy,indx,'r\*') ;

end

end

Mask = [ BW(indx-1,indy-1) BW(indx-1,indy) BW(indx-1,indy+1) ;...

BW(indx,indy-1 ) BW(indx,indy) BW(indx,indy+1 ) ;...

BW(indx+1,indy-1) BW(indx+1,indy) BW(indx+1,indy+1) ];

%% Condition 1

if Mask(2,3) == 0 && ( Mask(1,3) == 1 || Mask(1,2) == 1)

b1 = 1;

else

b1 = 0;

end

if Mask(1,2) == 0 && ( Mask(1,1) == 1 || Mask(2,1) == 1)

b2 = 1;

else

b2 = 0;

end

if Mask(2,1) == 0 && ( Mask(3,1) == 1 || Mask(3,2) == 1)

b3 = 1;

else

b3 = 0;

end

if Mask(3,2) == 0 && ( Mask(3,3) == 1 || Mask(2,3) == 1)

b4 = 1;

else

b4 = 0;

end

b = b1 + b2 + b3 + b4 ;

%% Condition 2

Mask = logical(Mask) ;

N1 = double( Mask(2,3) | Mask(1,3) ) + double( Mask(1,2) | Mask(1,1) ) + double( Mask(2,1) | Mask(3,1) ) + double( Mask(3,2) | Mask(3,3) );

N2 = double( Mask(1,3) | Mask(1,2) ) + double( Mask(1,1) | Mask(2,1) ) + double( Mask(3,1) | Mask(3,2) ) + double( Mask(3,3) | Mask(2,3) ) ;

NCondition2 = min(N1,N2) ;

%% Condition New3

Mask = logical(Mask) ;

Mask(1,1) = ~Mask(1,1) ;

NCondition3 = ( Mask(3,1) | Mask(3,2) | Mask(1,1) ) & Mask(2,1);

%% Judgement

if NCondition3 == 0 && NCondition2>=2 && NCondition2<=3 && b == 1

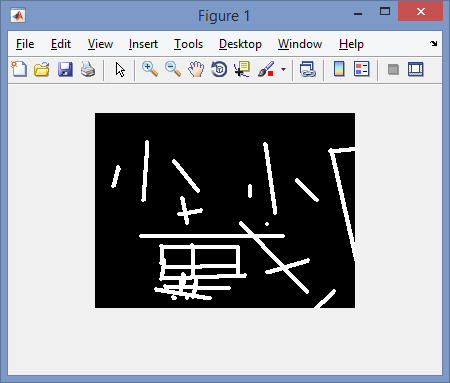
OutBW(indx,indy) = 0 ;

end

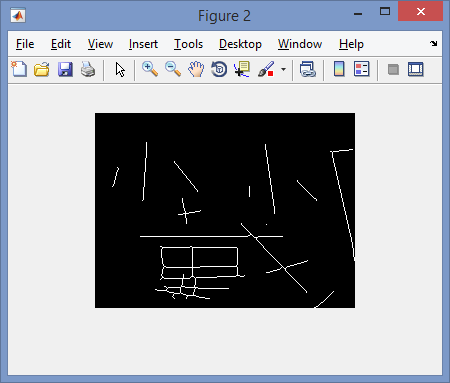
end

OutBW = OutBW([2:end-1],[2:end-1]) ;

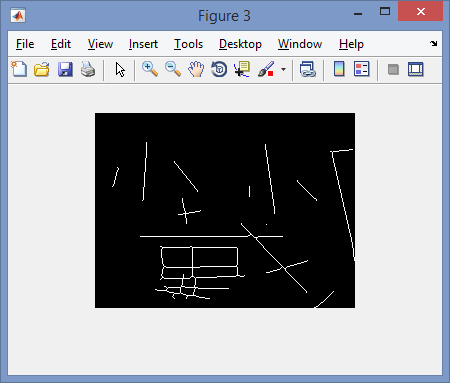
1. Output



*Gambar 1.0. Citra Asli*



*Gambar 1.1 Hasil akhir setelah proses thinning sebelum masuk black and white*



*Gambar 1.2, Citra Hasil Thinning*

1. Analisa

Thinning merupakan proses mengurangi intensitas atau pixel suatu objek dalam citra digital menjadi ukuran yang lebih kecil (objek citra menjadi seperti lidi, atau menjadi rangka (skeleton)). Thinning merupakan operasi morfologi yang digunakan untuk menghilangkan bagian depan piksel dari citra biner, yang dapat digunakan pada beberapa aplikasi namun sangat berguna untuk mengurangi citra menjadi sangat tipis (thin). Thinning merupakan bentuk pre-processing yang digunakan sebelum mengolah citra lebih lanjut. Output dari proses ini disebu “skeleton”. Thinning bertujuan untuk mengurangi bagian dari citra yang dianggap tidak perlu sehingga yang tersisa hanyalah citra yang penting atau dianggap esensial saja. Prosesnya sendiri adalah dengan cara mengurangi redudansi pada pixel yang memiliki nilai yang sama.